Universitat Autònoma de Barcelona

escola d'enginyeria

2164

Extracció del mapa de profunditat a partir d'una seqüència d'imatges

Memòria del Projecte Fi de Carrera
d'Enginyeria en Informàtica
realitzat per
Sergi Valls Company
i dirigit per
Gemma Sánchez Albaladejo

Bellaterra, 7 de setembre de 2010

El sotasignat, Gemma Sánchez Albaladejo

Professora de l'Escola d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en Sergi Valls Company.

I per tal que consti firma la present.

Signat: Gemma Sánchez Albaladejo

Bellaterra, 7 de setembre de 2010

Universitat Autònoma de Barcelona

escola d'enginyeria

El sotasignat, Jesús Rodríguez

de l'empresa, S3DTechnologies

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat en l'empresa sota la seva supervisió mitjançant conveni per a la realització de pràctiques en empreses i/o projecte final de carrera, que preveu el RD 1393/2007, de 29 d'octubre, firmat amb la Universitat Autònoma de Barcelona.

Així mateix, l'empresa en té coneixement i dóna el vist-i-plau al contingut que es detalla en aquesta memòria.

Signat: Jesús Rodríguez

Bellaterra, 7 de setembre de 2010

Índex

Capítol 1. Introducció.

Aquest projecte és fruit de l'interès d'una empresa per transformar de manera automàtica seqüències d'imatges de 2D a 3D. Durant les últimes dues dècades molts experts en visió per computador han buscat solucions al problema de la conversió automàtica de seqüències d'imatges de 2D a 3D. El pas automàtic de 2D a 3D estereoscòpic d'una seqüència d'imatges arbitrària no és actualment factible i requereix de processos manuals de creació de mapes de profunditat. La conversió a 3D és un procés cognitiu equivalent, per exemple, a la conversió de blanc i negre a color.

Tot i així, per algunes seqüències que satisfan certes condicions, sí que és possible realitzar la transformació. Les seqüències aptes per la reconstrucció 3D tenen en comú que la càmera està en moviment, el moviment és força pronunciat, es filma una escena estàtica i la qualitat de les imatges és bona (hi ha poc *motion blur*, resolució mitjana-alta...).

A partir de dos punts de vista d'una escena es pot percebre profunditat (per exemple el sistema de visió humà funciona d'aquesta forma mitjançant dos ulls). El moviment d'una càmera permet obtenir aquests diferents punts de vista d'una escena. El llibre de *Hartley i Ziserman* [1] explica amb detall tot el conjunt d'algorismes i fonaments matemàtics necessaris per tal de realitzar la conversió d'aquestes seqüències.

El projecte de final de carrera que presento resol les fases inicials d'aquest procés i té com a objectiu recuperar la informació de la posició i orientació de cada càmera d'una col·lecció d'imatges. Donat que l'empresa no té experiència sobre aquest tema, el projecte parteix des del començament.

Aquesta tecnologia es pot aplicar en altres àmbits de la visió per computador, des de la realitat augmentada fins a la robòtica.

1.1 – Anàlisi de viabilitat.

La branca de la visió per computador que es dedica a la recuperació de l'estructura d'una escena a partir de múltiples imatges (com a mínim dues) s'anomena *structure from motion*. Aquesta tecnologia permet deduir la posició a l'espai de la càmera per a cada imatge. Es fonamenta matemàticament en la geometria que relaciona múltiples vistes (geometria epipolar).

A l'actualitat existeixen *trackers 3D* molt potents com per exemple el *Boujou* [2] o el *PFTrack* [3] capaços de reconstruir les càmeres de tots els fotogrames d'una seqüència de vídeo i fins i tot extreure mapes de profunditat. Aquests programes demostren la viabilitat del projecte.

L’interès de l’empresa és confeccionar el seu propi producte, per tant no ens podem basar en els programes exsistents al marcat per tal d’assolir el nostre objectiu.

El *Boujou* és el *tracker* més estès a la indústria del cinema. La figura 1 mostra el *Boujou* integrant un model 3D a un fotograma d'un vídeo.

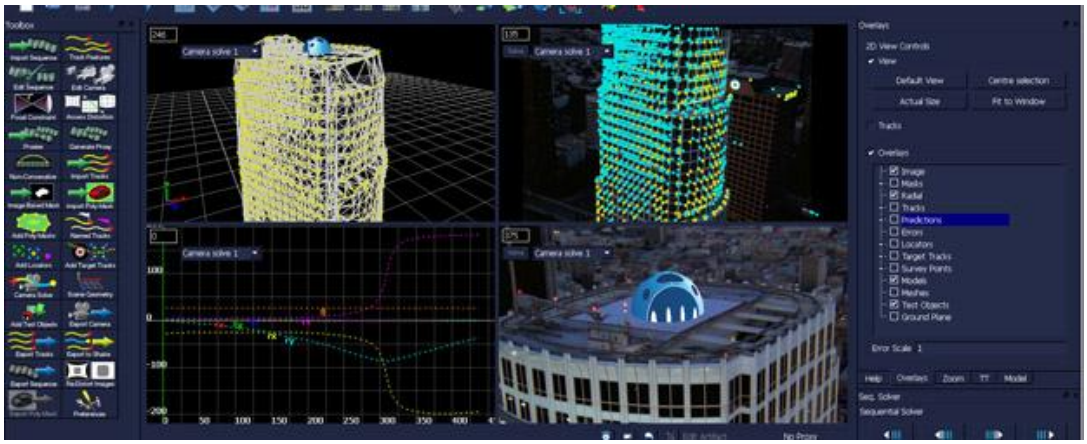


Fig. 1: *Boujou* analitzant un vídeo i integrant un model 3D en una superfície.

Photosynth [4] de *Microsoft* es un programa que processa col·leccions desordenades de fotos d'una escena (edifici representatiu, per exemple) reconstruint les càmeres i generant un núvol de punts tridimensional de l'escena. El programa utilitza després aquesta informació per mostrar l'àlbum de fotos de forma tridimensional. La figura 2 correspon a una captura del programa que combina una fotografia amb un núvol de punts.

Fig. 2: Captura del programa *Photosynth*

a) b)

Fig. 3: a) Robot *ASIMO* de *Honda*. b) *Structure from motion* d'una habitació en temps real basat en rectes.

El robot humanoide més avançat del món, *ASIMO* de *Honda* [5], utilitza un sofisticat sistema de *structure from motion* en temps real basat en rectes [6] per tal de poder percebre l'entorn. A la figura 3 es pot apreciar el mencionat robot juntament amb una imatge que mostra el *tracking* d'una habitació.

1.2 - Objectius.

L'objectiu de l'empresa és estudiar la viabilitat de reconstruir películes de 2D a 3D. Per tal de dimensionar correctament les tasques a realitzar i posar fites realistes fa falta tenir coneixement previ sobre el tema. Els primers dos mesos van consistir en l'elaboració d'un anàlisi de viabilitat i pla de treball per tal de ser capaç de proposar uns objectius viables i acotats al temps restant.

La idea general del procés consisteix a deduir les posicions relatives entre els diferents punts de vista de les imatges a partir de l'anàlisi de les similituts que hi ha entre elles. A partir d'aquí és factible triangular punts a l'espai i recuperar mapes de profunditat. La figura 4 mostra un diagrama d'aquest procés.

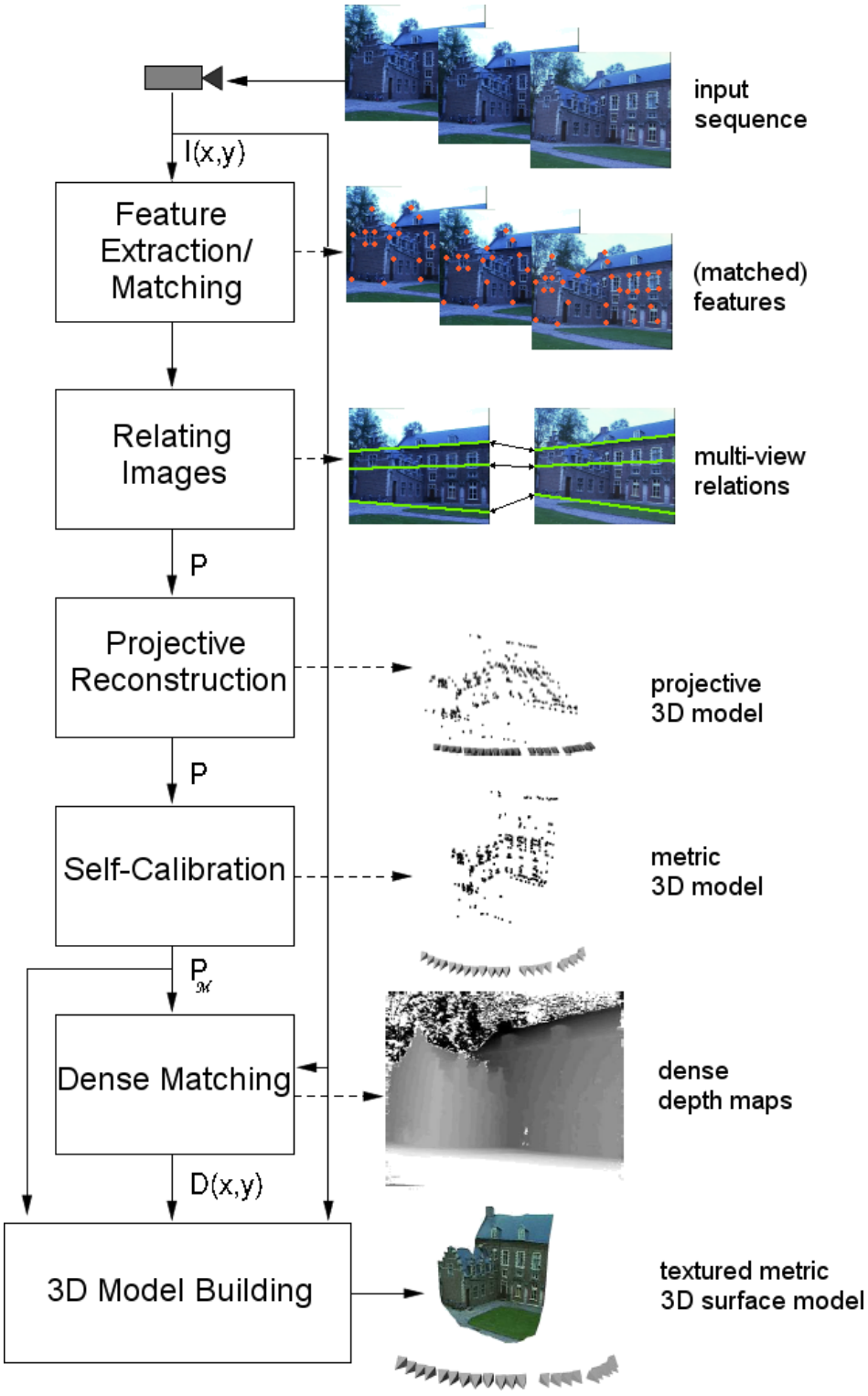


Fig. 4: Resum del procés de reconstrucció 3D a partir de seqüències d'imatges. Imatge extreta de [20]

La resolució del problema es pot resumir en les següents etapes:

- **Correspondències entre imatges:** consisteix a trobar punts en comú entre parells d'imatges. Si es tracta d'un vídeo, les correspondències s'obtenen mitjançant algorismes que segueixen els punts fotograma a fotograma analitzant el flux del moviment (*tracker 2D*). Si es tracta de col·leccions de fotos, es busquen punts significatius a totes les imatges, es calculen descriptors per cada punt, i es comparen els punts buscant els que més s'assemblen. Aquest últim cas és computacionalment més costós, però més senzill d'implementar. Implementar un *tracker 2D* robust i fiable podria ser un projecte independent.

- **Reconstrucció de càmeres:** consisteix a trobar la rotació i la translació de la càmera per a cada imatge. El resultat de la reconstrucció depèn fortament de certs paràmetres de la càmera (distància focal, mida del sensor, punt principal, coeficients de distorsió...). Per tal que la reconstrucció sigui proporcional a la realitat (mètrica, on la única incògnita és l'escala de la reconstrucció) fan falta els paràmetres. Si no es disposen de tals paràmetres només es pot obtenir una reconstrucció projectiva que pot no assemblar-se gens a la realitat. Més endavant explicaré amb detall còm afecten els paràmetres a la reconstrucció, concretament el més significatiu, la distància focal. Actualitzar una reconstrucció projectiva a una reconstrucció mètrica s'anomena reconstrucció no calibrada. Exsisteixen programes com el *Boujou* treballen de forma no calibrada. Els mètodes d'autocalibració són complexes d'entendre i implementar. Al final del procés de reconstrucció s'aplica un algoritme anomenat *Bundle Adjustment*. És un algorisme que s'encarrega de minimitzar l'error comès a la

fase de detecció de correspondències a partir dels punts reconstruïts. Modifica tota l'estructura, des de les càmeres fins als punts 2D i 3D, de forma iterativa. És capaç també de deduir i ajustar els paràmetres interns de les càmeres.

- **Rectificació d'imatges:** pas previ al càlcul del mapa de profunditat que consisteix en aplicar una transformació a les imatges que fa que els píxels de les línies horitzontals coincideixin. D'aquesta manera trobar el mapa de profunditat passa de ser un problema bidimensional a un d'unidimensional.

- **Obtenció del mapa de profunditat:** consisteix a calcular la profunditat de cada píxel d'una imatge a partir de dues imatges rectificades. Solen ser algorismes molt sofisticats que analitzen la disparitat entre dues imatges a partir de files de píxels.

Partint d'aquesta visió general, el meu projecte s'ha centrat en la **reconstrucció de càmeres calibrades a partir de col·leccions de fotografies** que suposarem preses per la mateixa càmera. La reconstrucció de càmeres calibrades conté tots els elements necessaris per aprendre els conceptes de la matèria i, en un futur, poder abordar la reconstrucció no calibrada i processament de vídeos. Reconstruir vídeos és més complicat ja que es necessita un *tracker 2D* robust, i la reconstrucció de càmeres no és tant directa, ja que entre dos fotogrames d'un vídeo hi ha tant poc moviment que de vegades no es suficient per reconstruir i triangular.

1.3 - Requisits.

Pel desenvolupament del projecte és suficient un ordinador convencional. En aquest cas s'ha usat un portàtil amb processador *Intel core 2 duo* a 2.1 GHz amb 2GB de RAM.

Les imatges no cal que siguin generades per càmeres especials d'altes prestacions. Pel projecte s'han fet servir càmeres d'ús domèstic. La gran majoria de càmeres convencionals graven, en el propi fitxer JPEG, informació extra com per exemple la marca de la càmera, model i fins i tot la distància focal i mida del sensor. En cas de no disposar de la mida del sensor, buscant les especificacions tècniques del model de la càmera es pot trobar ràpidament. Aquestes dades permetran, en principi, realitzar una reconstrucció mètrica.

1.4 - Estructura de la memòria.

Aquesta memòria es composa dels següents capítols:

- El segon capítol és un breu resum dels conceptes bàsics necessaris pel desenvolupament del projecte. Gran part del temps del projecte ha estat dedicat a l'adquisició d'aquests coneixements. El capítol té com a objectiu explicar com es modela matemàticament una càmera, com obtenir la posició relativa entre elles, quins problemes comporta, quins elements intervenen a la reconstrucció, etc. El capítol no entra en detall en cap dels algorismes ni mostra cap demostració matemàtica. Tots els algorismes i demostracions es poden trobar documentats amb detall al llibre de *Hartley i Ziserman* [1].

- El tercer capítol explica quins són els mètodes que es fan servir per tal de relacionar les diferents fotografies. Explica els avantatges i inconvenients de cada un i en quins casos és més convenient usar un o altre.

- El quart capítol descriu les eines i les etapes de desenvolupament del programa. Es centra en la part d'enginyeria i descriu com s'ha organitzat i dissenyat el projecte.

- El cinquè capítol mostra els resultats obtinguts.

- Al sisè capítol s'exposen les conclusions i quins serien els següents passos per tal de millorar el programa i ampliar el ventall de funcionalitats.

- Finalment, els últims apartats corresponen al glossari i la bibliografia.

Capítol 2. Reconstrucció de càmeres.

La reconstrucció de càmeres requereix entendre amb profunditat com es formen les imatges en una càmera projectiva. A partir d'aquí podem modelar matemàticament la geometria que relaciona les diferents càmeres d'una escena per tal d'obtenir la posició i orientació del dispositiu en el moment que va realitzar la fotografia. Després cal sotmetre la reconstrucció a processos d'optimització per tal de refinar els resultats.

2.1 - La càmera projectiva.

Una càmera és un dispositiu que tradueix coordenades del món 3D a coordenades d'una imatge 2D. La figura 5 mostra un esquema del funcionament d'una càmera projectiva i com es projecta un punt 3D a un pla.

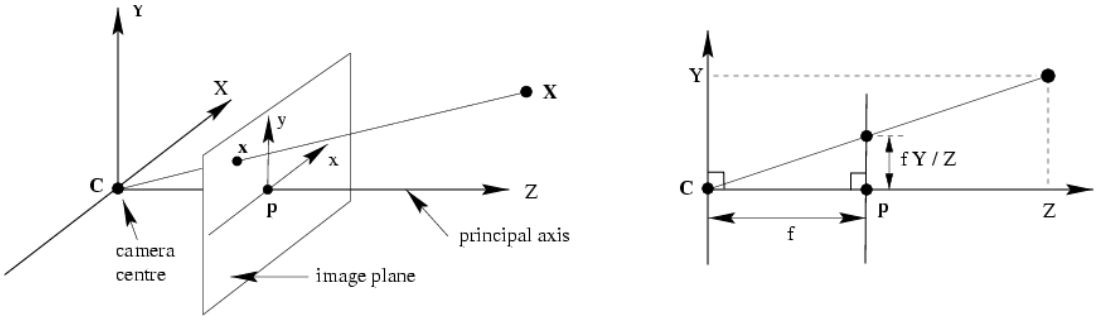


Fig. 5: Esquema del model d'una càmera projectiva. Imatge extreta de [1].

La càmera projectiva es pot modelar matemàticament amb la següent matriu:

$$K = \begin{pmatrix} \frac{f}{p_x} & \tan\alpha \frac{f}{p_y} & c_x \\ 0 & \frac{f}{p_y} & c_y \\ 0 & 0 & 1 \end{pmatrix}$$

f: distància entre el centre de projecció i el pla retinal.

p_x,p_y: amplada i alçada dels píxels.

c_x,c_y: coordenades del punt principal (punt *p* a la figura 4).

α: angle d'obliqüitat del píxel.

En la gran majoria d'ocasions es pot suposar que el píxel és quadrat i la seva obliqüitat és zero. En tal cas, la matriu de cal·libració queda simplificada de la següent manera:

$$K = \begin{pmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{pmatrix}$$

Multiplicant la matriu K per un punt 3D obtenim les coordenades del punt a l'espai projectiu. Per tal de trobar les coordenades (x,y) del punt projectat, fa falta dividir el punt per la tercera component (coordenades homogènies), obtenint un vector de la forma (x,y,1).

En general, la projecció d'un punt tridimensional a una càmera donades una orientació i una translació es defineix com:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = K(R^T \; -R^T t) \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$p = \begin{pmatrix} x/z \\ y/z \end{pmatrix}$$

R: matriu de rotació (dimensions 3x3).

t: vector de translació (dimensions 3x1).

p: coordenades del punt 3D projectades a la càmera.

2.2 - Ambigüitat projectiva.

Els paràmetres interns de la càmera són necessaris per tal d'obtenir una reconstrucció mètrica. La reconstrucció mètrica ens garanteix que les proporcions d'escena són les originals, però el que no es pot arribar a determinar és la mida original de l'escena. En altres paraules, no sabem si l'objecte a reconstruir és molt petit i la càmera l'enregistra a poca

distància, o si l'objecte és molt gran i la càmera el capta des de lluny. La figura 6 mostra aquest fenomen.

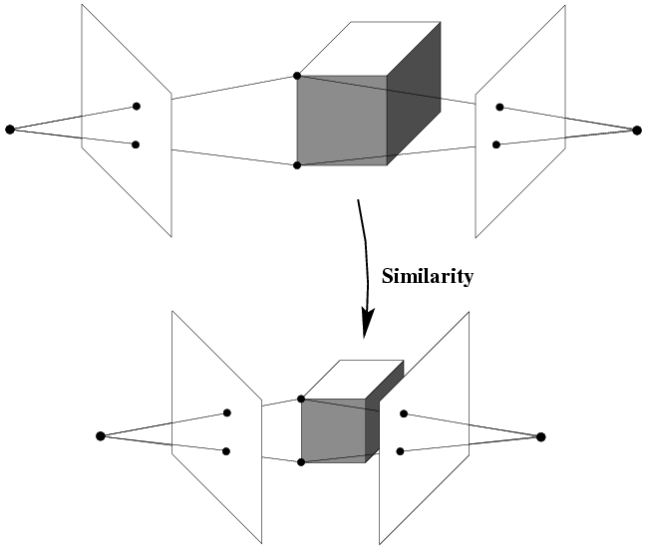


Fig. 6: Ambigüitat a l'escala de la reconstrucció. També s'anomena reconstrucció "similar". Imatge extreta de [1].

Quan no disposem dels paràmetres interns de la càmera l'ambigüitat augmenta. En aquest cas s'anomena ambigüitat projectiva. Aquesta ambigüitat no respecta les proporcions i els angles de la figura original. Usar una distància focal incorrecta provoca que el punt escollit al triangular no sigui l'adequat. La reconstrucció és coherent només des d'un punt de vista projectiu. La figura 7 mostra què passa a la reconstrucció quan canviem la focal de les càmeres.

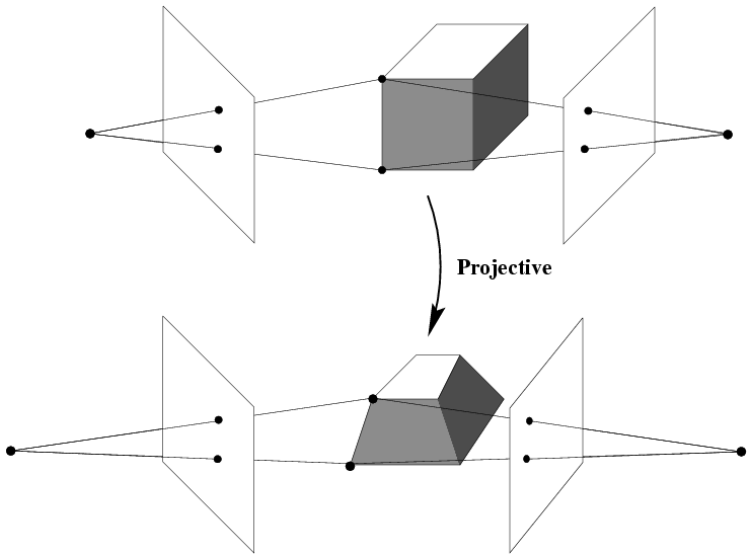


Fig. 7: Triangulació de punts entre dues vistes. A sobre, les càmeres tenen la distància focal adequada i la reconstrucció és mètrica. A sota, la distància focal no és la correcta i apareix una reconstrucció deformada. Imatge extreta de [1].

2.3 - La geometria epipolar.

El primer que se'ns pot passar pel cap si volem desfer una projecció és fer el procés invers a projectar. Si tracem una recta que passi pel centre de projecció d'una càmera i per un punt concret de la imatge (procés invers a projectar), obtindrem una recta que passa pel punt 3D original, però no sabem quin és exactament. Si disposéssim d'un altre punt de vista i féssim el mateix pel mateix punt, podríem triangular-lo a l'espai.

La projecció d'aquesta recta a una altra vista s'anomena línia epipolar. El punt 3D original es projecta en algun punt d'aquesta línia en l'altra vista. La figura 8 mostra com es forma la línia epipolar l' a la vista de la dreta.

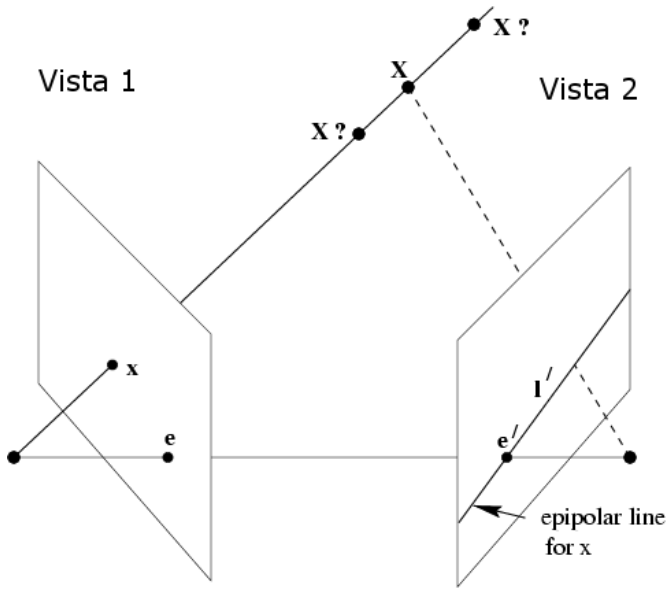


Fig. 8: Esquema que mostra com es relaciona un punt *x* d'una vista 1 amb una recta *l′* en una altra vista 2. També mostra com el punt 3D *X* original cau en algun punt d'aquesta recta *l′*. Imatge extreta de [1].

La geometria epipolar és la geometria que relaciona les diferents vistes d'una escena obtingudes a partir de càmeres projectives. Essencialment es basa en la intersecció dels plans de projecció de les càmeres amb els (infinits) plans tangents a la recta que uneix els dos centres de projecció de les càmeres (línia base).

La intersecció de la línia base amb els plans de projecció s'anomenen epipols. La figura 9 mostra esquemàticament els epipols *e* i *e′* i alguns plans tangents a la línia base.

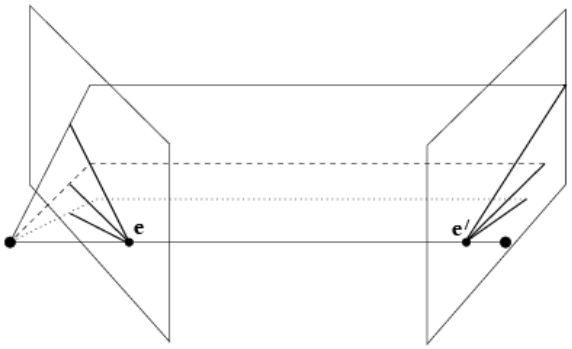


Fig. 9: Representació gràfica de la geometria epipolar entre dues vistes. Imatge extreta de [1].

La propietat més interessant d'aquesta geometria és que els punts de la recta formada per la intersecció d'un pla de projecció amb un pla P tangent a la línia base coincideixen amb els punts de la recta formada per la intersecció del pla de l'altra càmera amb el mateix pla P. La figura 10 permet apreciar aquesta propietat.

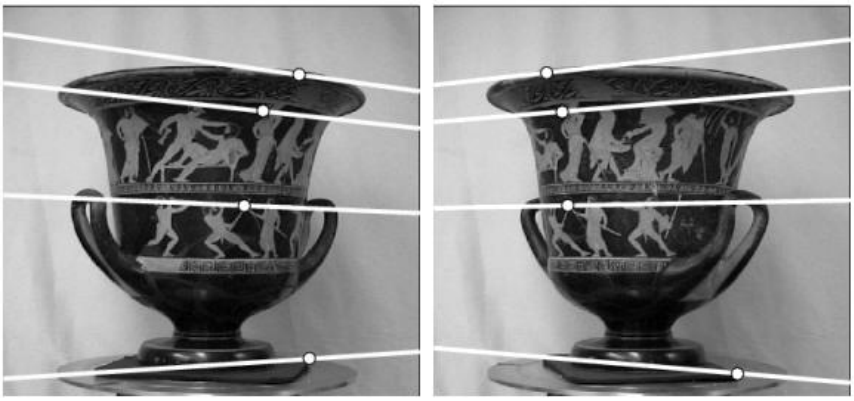


Fig. 10: Dues vistes d'un gerro amb algunes rectes epipolars. Imatge extreta de [1].

Com es pot apreciar, els punts de les rectes de la vista de l'esquerra corresponen amb els punts de les rectes de la vista de la dreta.

2.3.1 - La matriu Fonamental

La matriu Fonamental és la representació algebraica de la geometria epipolar de dues vistes. Siguin m i m' punts corresponents entre dues imatges, la matriu Fonamental F compleix la següent igualtat:

$$m'^T F m = 0$$

La matriu Fonamental es pot estimar a partir de punts corresponents resolent la equació anterior. Per computar F són suficients 8 parells de punts corresponents per un algorisme lineal. És una matriu de rang 2 (transforma punts 2D d'una vista a rectes epipolars a l'altra vista).

La matriu Fonamental captura la relació projectiva que hi ha entre dues càmeres. Això significa que una mateixa matriu Fonamental representa la configuració d'infinits parells de càmeres que difereixen tan sols per una transformació projectiva (ambigüitat projectiva). Entre un parell de càmeres només existeix una sola matriu Fonamental.

La matriu Fonamental, un cop calculada, és de gran ajuda a l'hora de cercar punts corresponents entre imatges ja que proporciona una guia sobre on es troba un punt a l'altre imatge (línia epipolar).

El tensor trifocal realitza la mateixa funció que la matriu Fonamental però per tres vistes enlloc de dues. S'utilitza sobretot durant la reconstrucció de vídeos i per tant serà objecte d'estudi en un futur, però no durant aquest projecte de fi de carrera per raons de temps. Usant la

geometria de tres vistes s'obtenen reconstruccions més robustes i precises.

Existeix el cas general de la geometria de N vistes, però la complexitat de les relacions geomètriques fa que més de quatre vistes sigui impracticable i innecessari.

2.3.2 - La matriu Essencial.

La matriu essencial és l'especialització de la matriu Fonamental que elimina l'ambigüitat projectiva. Es calcula de la següent manera:

$$E = K'^{T} F K$$

K: matriu de cal·libració de la primera càmera.

K': matriu de cal·libració de la segona càmera.

F: matriu Fonamental.

2.4 - Obtenció de la posició relativa entre dues càmeres.

A partir de dues vistes i de la matriu Essencial que les relaciona podem trobar el desplaçament de la segona càmera respecta la primera. Suposant un parell de càmeres amb paràmetres extrínsecs P i P' essent $P = (I|0)$ (sense rotació ni translació) i una matriu essencial E, la matriu P' es calcula de la següent manera:

Primer cal fer la descomposició SVD (*singular value decomposition*) de la matriu E:

$$SVD(E) = U diag(1,1,0) V^T$$

A partir de la descomposició de E:

$$P' = \left(UWV^T | +u3\right)$$

$$P' = \left(UWV^T | -u3\right)$$

$$P' = \left(UW^TV^T | +u3\right)$$

$$P' = \left(UW^TV^T | -u3\right)$$

Sent $u3$ la tercera columna de U i $W = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$.

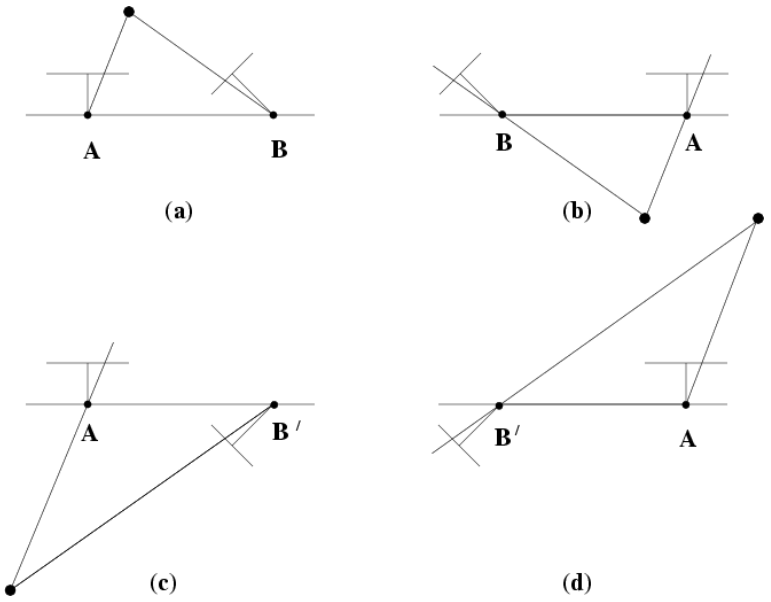


Fig. 11: Interpretació geomètrica de les quatre possibles reconstruccions de la càmera. Imatge extreta de [1].

Com es pot observar, existeixen quatre solucions i només una d'elles es vàlida. La solució vàlida és aquella que reconstrueix els punts davant de les dues càmeres. La figura 11 il·lustra les quatre possibilitats.

2.5 - Obtenir la posició a partir de la reconstrucció.

Una altra forma d'obtenir la orientació d'una càmera és a partir de correspondències entre punts 3D amb punts 2D d'una imatge. Projectant els punts 3D a la càmera i mesurant l'error entre els corresponents punts 2D (error de reprojecció), podem obtenir una configuració de la càmera que minimitzi aquest error en funció de la posició i la rotació. Aquest mètode és conegut com DLT (*Direct Linear Transform*).

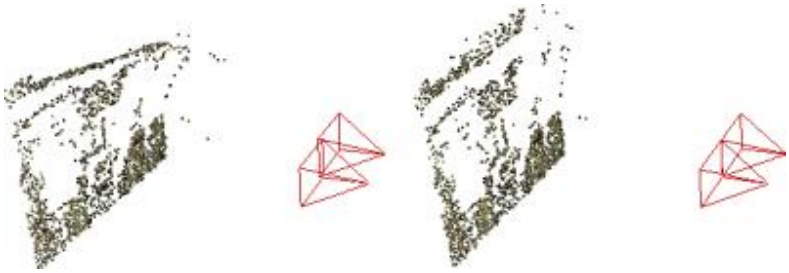
Aquest mètode és útil quan es disposen de punts ja reconstruïts a partir d'altres d'imatges i la nova imatge és força similar a les anteriors. D'aquesta manera podem reconstruir la càmera fins i tot en els casos en que la proximitat entre càmeres impedeix una bona triangulació, com podria ser en fotogrames consecutius en seqüències de vídeo.

2.6 - *Bundle Adjustment.*

Bundle Adjustment és un algorisme d'optimització que s'aplica al final de la reconstrucció. L'algorisme refina simultàniament cada punt reconstruït, posició i orientació de les càmeres, paràmetres interns de les càmeres, basant-se en l'error de reprojecció dels punts reconstruïts. Per minimitzar l'error de reprojecció en funció dels paràmetres intrínsecs i extrínsecs de la càmera, els punts projectats i els punts reconstruïts es

fa servir el mètode de *Levenberg–Marquardt*. Aquesta optimització només és possible si es pressuposa que l'error global de la reconstrucció deriva de l'error comès a la fase de detecció de punts, és uniforme i té una distribució normal.

a)



b) c)

Fig. 12: a) Imatges d'entrada. b) Reconstrucció no optimitzada. c) Reconstrucció optimitzada.

A la figura 12 es pot observar la diferència entre una reconstrucció optimitzada i una no optimitzada. Les imatges d'entrada corresponen a una façana generada per ordinador. El moviment de la càmera és totalment horitzontal. Les piràmides corresponen a les càmeres sent la

base el pla retinal. La reconstrucció de l'esquerra no està optimitzada i es pot apreciar que la zona amb menys densitat de punts es veu afectada i deformada. Les càmeres tampoc corresponen a un moviment totalment horitzontal. En canvi, a la reconstrucció optimitzada mitjançant *Bundle Adjusment* tots aquests problemes no apareixen.

2.7 - Reconstrucció de múltiples vistes.

Com hem vist fins ara, és senzill reconstruir les càmeres de dues vistes a partir de punts corresponents i els paràmetres intrínsecs de les càmeres i obtenir un núvol de punts tridimensional de l'escena a partir de la triangulació de les correspondències. El següent pas és reconstruir N vistes d'una escena sense haver de recórrer a la geometria de N vistes que, com ja hem comentat, és impracticable.

Suposem tres vistes d'una escena, vistes 1, 2 i 3. Una primera aproximació consistiria en situar la vista 1 a l'origen i reconstruir la vista 2 mitjançant la reconstrucció de dues vistes. Aleshores, trobaríem la posició de la vista 3 respecte la vista 2 aplicant la reconstrucció de dues vistes entre 2 i 3. Per acabar de situar la vista 3, aplicaríem un canvi de base per situar la càmera respecte l'origen (vista 1). Aquesta primera aproximació no funciona del tot bé, ja que la reconstrucció té ambigüitat d'escala. Ens trobem doncs el problema que l'escala entre les vistes 1 i 2 no és la mateixa que entre les vistes 2 i 3. En conseqüència, el núvol de punts que formen les vistes 1 i 2 no encaixa amb el núvol de punts que formen les vistes 2 i 3 tal i com es pot apreciar a la figura 13. Per tal de que la vista 3 sigui coherent amb la vista 1, fa falta que la vista 3 es reconstrueixi a partir d'informació de les vistes 1 i 2.

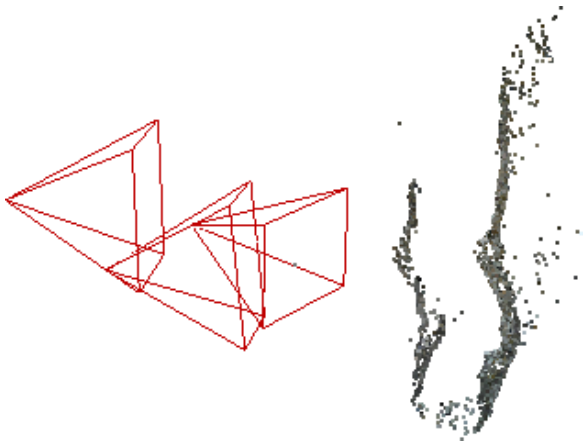


Fig. 13: Reconstrucció resultant al encadenar reconstruccions entre parelles de vistes.

Una opció seria usar el tensor trifocal i per a cada nova vista reconstruir-la a partir de dues altres ja reconstruïdes. Aquest mètode s'utilitza principalment en seqüències de vídeo, però en col·leccions de fotografies es pot anar més enllà i usar totes les vistes possibles per tal d'obtenir més precisió. El mètode que faig servir en aquest projecte està inspirat en el mètode descrit a [15] i combina la reconstrucció de dues vistes, *DLT* i *Bundle Adjustment*.

Primer es seleccionen dues vistes que tinguin un alt nombre de correspondències però que a la vegada siguin força distants i es realitza una primera reconstrucció de dues vistes. Les correspondències es triangulen i s'optimitza l'estructura mitjançant *Bundle Adjustment.* Es important partir d'una bona primera reconstrucció de dues vistes.

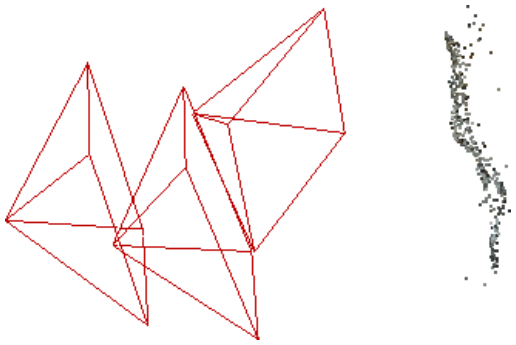


Fig. 14: Reconstrucció coherent amb totes les vistes.

Per tal d'afegir més vistes a la reconstrucció, es selecciona una nova vista que tingui correspondències amb les vistes ja reconstruïdes. D'aquestes correspondències, es seleccionen les que ja estan triangulades i es reconstrueix la posició de la càmera mitjançant *DLT*. A continuació es trianguleN la resta de correspondències i s'optimitza l'estructura. D'aquesta manera ens assegurem que cada nova càmera és consistent amb l'estructura global de la reconstrucció. La figura 14 correspon a la reconstrucció de les mateixes tres vistes que la figura 13 usant el procés anterior.

Capítol 3. Correspondències entre imatges.

El capítol anterior explica com reconstruir les càmeres a partir de punts corresponents. Fa falta llavors trobar un sistema robust d'obtenció de punts i correspondències entre imatges. Una possibilitat és trobar punts i correspondències manualment. Aquesta possibilitat queda descartada ja que manualment es comet força error i per un gran nombre de fotografies es converteix en una tasca impracticable. En aquest capítol explicaré com se solen trobar aquests punts i correspondències de forma automàtica i quins algorismes es fan servir.

3.1 - Mètodes de detector-descriptor.

Aquests algorismes consisteixen en dues parts: detecció de punts característics en una imatge i càlcul de descriptor dels punts.

L'algorisme de detecció de punts troba píxels a la imatge que considera diferenciables als seus veïns analitzant canvis de contrast pronunciats. Cada algorisme realitza aquesta tasca de formes diferents:

- L'algorisme de *Shi i Tomasi* [7] és un detector de punts que busca mínims locals a regions d'imatges a partir del gradient.

- Algorismes com *SIFT* [8] o *SURF* [9] pretenen que el punt sigui detectable a diferents escales (invariància en escala) i computen una piràmide formada per còpies de la imatge a diferents escales. La figura 15 mostra els punts trobats després d'aplicar *SIFT* a una imatge.

- L'algorisme *FAST* [10] prioritza l'eficiència i l'aconsegueix mitjançant aprenentatge. No té en compte diferents escales de la imatge.

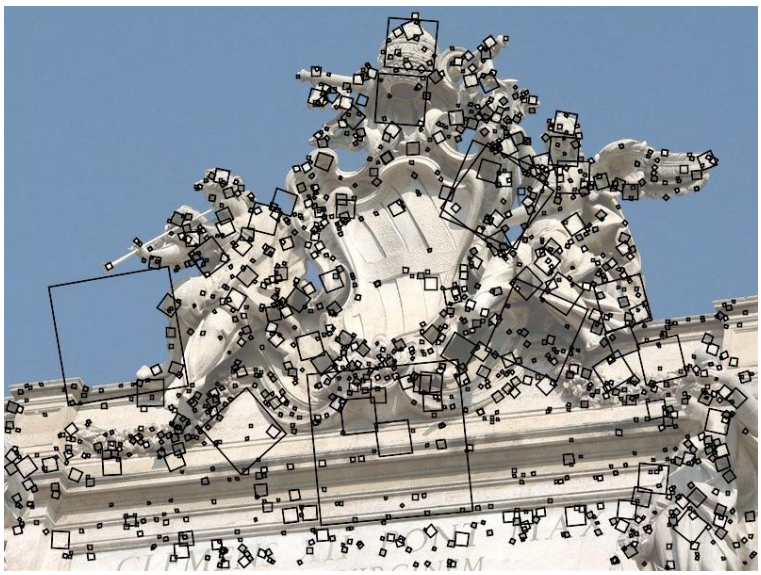


Fig. 15: Punts característics detectats amb *SIFT*.

El descriptor és un vector que resumeix les característiques d'un punt per tal de poder comparar-lo amb altres punts mitjançant la distància euclidiana. Aquests descriptors poden ser emmagatzemats en bases de dades per tal de ser consultats més tard a l'hora de fer cerques de patrons o objectes.

Existeixen varis algorismes de càlcul de descriptors (*Daisy descriptor*, *STAR descriptor...*). En aquest projecte s'han utilitzat els descriptors que incorporen els algorismes *SIFT* i *SURF*, ja que són descriptors invariants en escala, rotació, lluminositat i transformacions afins, sent ideals per trobar correspondències entre diferents vistes d'un objecte. L'algorisme SIFT s'utilitza a l'aplicació *Photosynth.*

3.2 - Anàlisi del moviment.

Suposant una seqüència de vídeo on els fotogrames no presenten variacions gaire pronunciades no és agosarat pensar que un punt en un fotograma es localitza en un lloc pròxim en el següent. El conjunt d'algorismes que analitzen les variacions entre fotogrames s'anomenen algorismes d'anàlisi de flux. En aquests mètodes la cerca de correspondències està implícita a la pròpia naturalesa de l'algorisme.

L'algorisme més robust que presenta millors resultats s'anomena *Lucas-Kanade optical flow* [11]. Tot i ser el més robust, sempre apareixen falsos positius tal i com es pot comprovar a la figura 16.

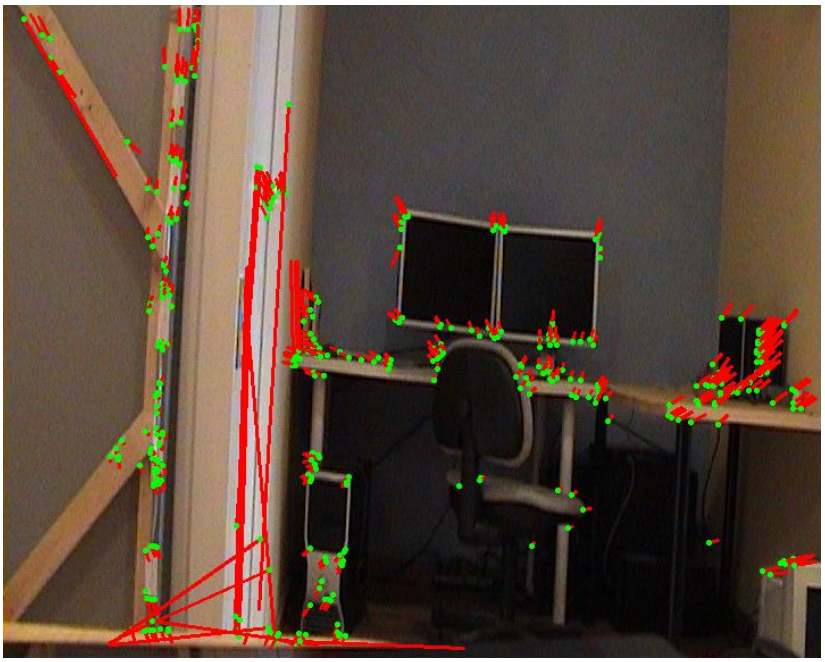


Fig. 16: *Lucas-Kanade optical flow*. Els punts han estat inicialitzats mitjançant *FAST*. Les rectes vermelles marquen la trajectòria que ha seguit el punt des del fotograma anterior.

3.3 - Cerca de correspondències.

Un cop obtinguts els descriptors dels punts de dues imatges fa falta comparar-los entre ells i trobar el veí més pròxim de cada un a l'altre imatge. Aquest procés sol ser de complexitat lineal ja que cada descriptor s'ha de comparar amb tots els de l'altre imatge. Per tal d'evitar la complexitat lineal es poden usar arbres de clústers indexats [12] que garanteixen una solució sub-òptima prou bona que proporciona una reducció de varis ordres de magnitud del temps de cerca.

Fig. 17: Correspondències entre dues imatges mitjançant descriptors SIFT. Imatge extreta de [16].

Per trobar una correspondència fa falta trobar la distància del veí més pròxim (d1) i la distància del segon veí més pròxim (d2). A partir d'aquestes dues distàncies es calcula la relació que hi ha entre elles (d1/d2). Si el quocient de les dues distàncies es major a una tolerància especificada (sol ser un valor entre 0.4 i 0.8) la correspondència es considera errònia i es descarta, si es menor, es considera vàlida.

La figura 17 mostra un exemple de cerca de correspondències entre dues imatges mitjançant la comparació de descriptors. Es pot apreciar que, com en la majoria de casos, apareixen alguns falsos positius.

3.4 - Filtratge de falsos positius a partir de restriccions geomètriques.

Una de les tasques més importants durant la reconstrucció és saber detectar i gestionar els falsos positius. Fan falta molt poques falses correspondències per tal d'esbiaixar completament una reconstrucció. Per tal de filtrar les correspondències obtingudes i descartar les falses s'utilitzen les restriccions geomètriques de l'escena.

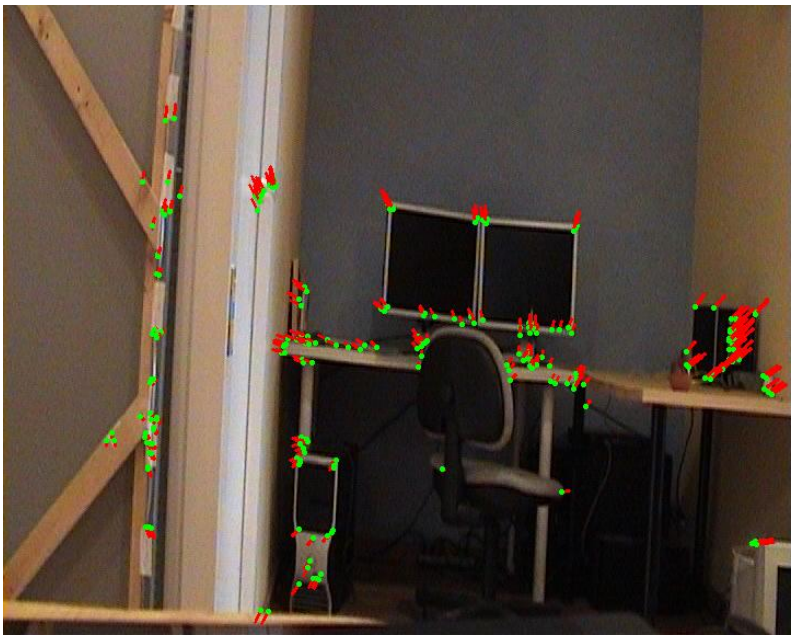


Fig. 18: Filtratge de correspondències a partir de la matriu Fonamental.

El càlcul de la matriu Fonamental es realitza de forma iterativa introduint un parell de punts corresponents en cada iteració. Sabem doncs que un punt en una imatge correspon a una recta a l'altra imatge. Al introduir un nou punt podem comprovar si la seva parella és consistent amb la geometria calculada fins al moment. Si la distància del

punt a la recta epipolar que li correspon és inferior a una tolerància donada (normalment menor a 3 píxels) el punt es considera vàlid i s'utilitza per refinar la matriu Fonamental. Si la distància és superior a la tolerància el punt es descarta. Aquest algorisme iteratiu de detecció de soroll en mostres s'anomena *RANSAC* (*RANdom SAmple Consensus* [13]).

Com es pot apreciar a la figura 18, la majoria de falsos positius de la figura 16 han desaparegut després del filtratge.

Capítol 4. Desenvolupament del programa.

Per tal de desenvolupar una aplicació d'aquest tipus partint del desconeixement de la matèria comporta força problemes de disseny degut a que no es disposa una visió global del problema i de tots els detalls que comporta. Cal començar per la base, afegir millores i funcionalitats de forma iterativa visualitzant i validant minuciosament els resultats a cada etapa. Aquest fet ha comportat canvis de disseny radicals entre versions creant varis prototips. En aquest capítol s'exposa com s'ha abordat el projecte des del punt de vista del desenvolupament.

4.1 - Anàlisi de requeriments.

El programa ha de ser capaç de carregar una col·lecció d'imatges fetes amb la mateixa càmera (amb els mateixos paràmetres intrínsecs), visualitzar-les, i realitzar una reconstrucció mètrica mitjançant la tècnica

descrita al llibre de *Hartley i Ziserman* [1]. La cerca de punts i correspondències ha de ser automàtica.

4.2 - Planificació.

El projecte s'ha dimensionat en set mesos. Les tàsques generals a realitzar són les següents: estudi de viabilitat, estudi de l'estat de l'art, programació, *test* i elaboració de la memòria i presentació.

El diagrama de Gantt de la planificació es mostra a la figura 19. Aquesta planificació es va realitzar després d'haver adquirit una petita base de coneixements sobre la matèria. Al no tenir una visió global i detallada del problema no va ser possible dividir cada tasca en diferents subtàsques més concretes. Tot i ser una planificació de caràcter molt general els terminis previstos s'han complert amb èxit.

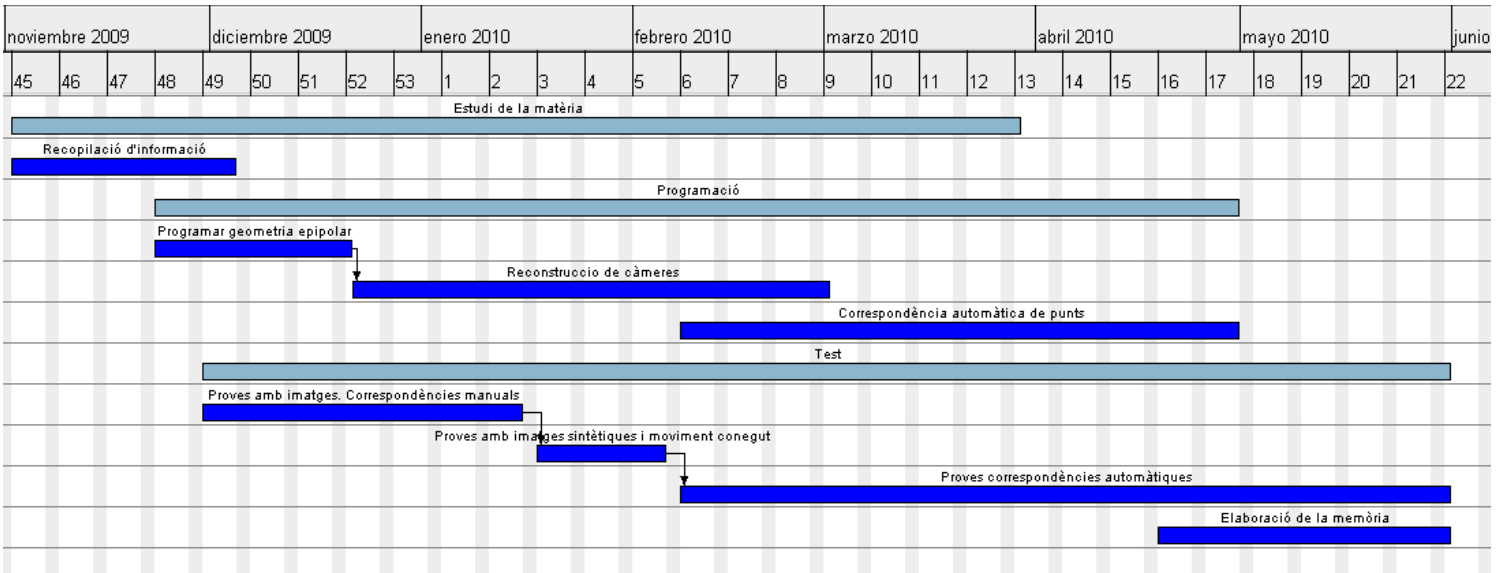


Fig. 19: Diagrama de Gantt.

L’estudi de l’estat de l’art es prolonga durant tot el projecte ja que el desenvolupament permet aprofundir en la matèria. La programació consisteix en resoldre la geometria epipolar, reconstrucció de càmeres i correspondència automàtica de punts. El *test* es realitza conjuntament amb la programació durant tot el desenvolupament. Consisteix en realitzar proves amb imatges i correspondències manuals, proves amb imatges sintètiques i moviment conegut i proves amb correspondències automàtiques.

El projecte té una duració de 30 setmanes a mitja jornada (20 hores setmanals), suposant un total de 600 hores. Suposant un preu per hora de 8€ el projecte té un cost total de 4800€.

4.3 – Dissenys del programa.

4.3.1 – Reconstrucció calibrada de dues vistes.

El primer disseny va consistir en resoldre el cas més bàsic: la reconstrucció calibrada de dues vistes. El prototip consistia en una interfície molt rudimentària programada en *GLUT* i *OpenCV*. El programa permetia visualitzar dues fotografies, introduir punts manualment, visualitzar les rectes epipolars i la reconstrucció obtinguda. La reconstrucció mostrava les quatre possibles solucions de la càmera.

L'objectiu d'aquest disseny era familiaritzar-se amb part matemàtica del problema sense haver de perdre temps programant una bona interfície gràfica i cerca automàtica de punts. La figura 20 mostra l’aspecte del programa. A la figura es poden apreciar les dues rotacions obtingudes de la càmera. Degut l’escassetat de punts i l’error comès al posar-los manualment les rectes epipolars no encaixen del tot bé.

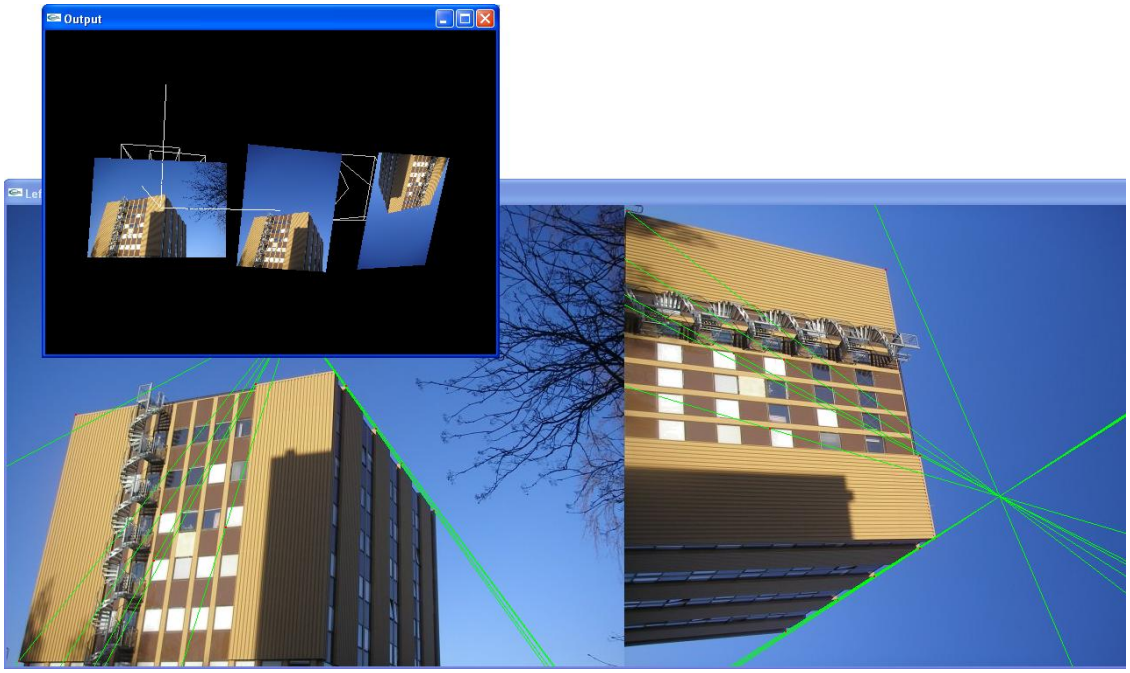


Fig. 20: Captura de la primera versió del programa.

4.3.2 – Correspondències automàtiques.

Un cop resolta la part clau del problema es va prosseguir a un segon disseny. El segon disseny incorporava cerca de punts i correspondències automàtiques mitjançant *SURF* i *FLANN*. Es va implementar l'obtenció de la solució única per tal de descartar les tres solucions incorrectes. Es va incorporar la generació del núvol de punts de l'escena.

La interfície gràfica es va reimplementar completament usant *WxWidgets*. Gràcies a la llibreria es va poder programar un bon visor d'imatges que permetia visualitzar les fotografies més grans adaptant la seva mida a la de la finestra, fer zoom a qualsevol punt de la imatge i desplaçar-la mitjançant el ratolí. El visor permetia també mostrar els

punts trobats, les correspondències bones i les rectes epipolars que, mitjançant el zoom, permetien visualitzar els epipols. La figura 21 mostra la nova interfície. Els epipols es localitzen a dins de les imatges ja que corresponen a un desplaçament frontal.

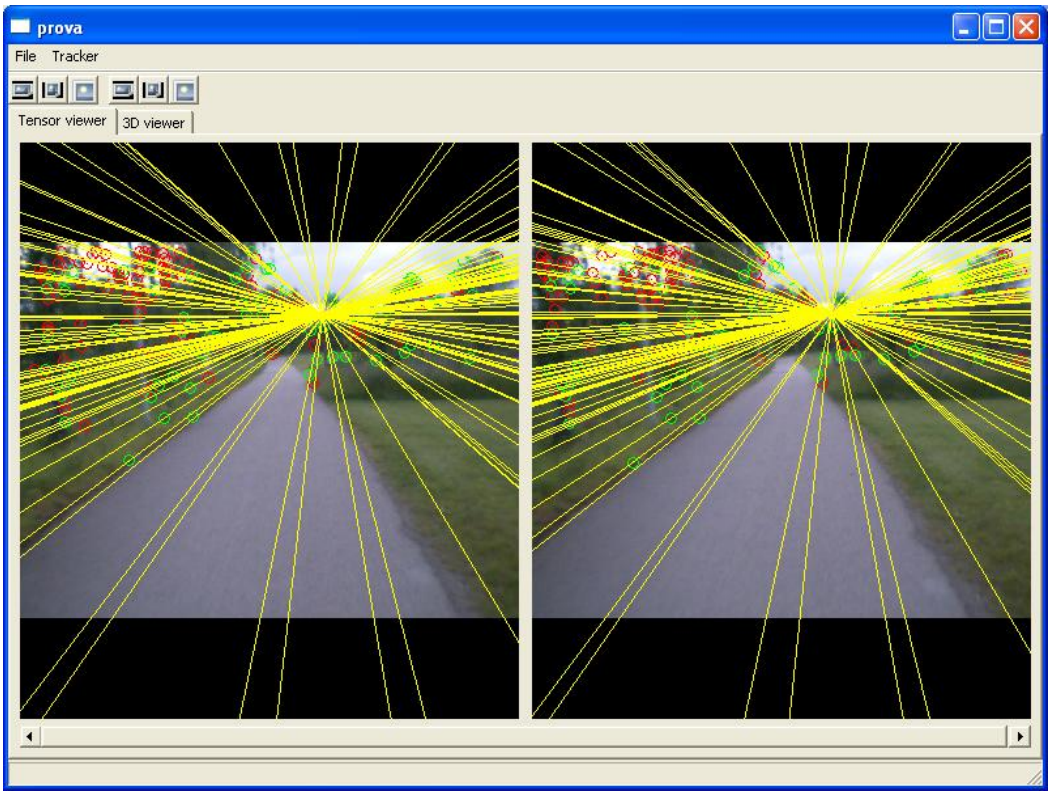


Fig. 21: Captura de la segona versió del programa.

4.3.3 – Reconstrucció de més de dues càmeres i *Bundle Adjustment*.

El tercer disseny implementava la reconstrucció de més de dues càmeres a base d'encadenar reconstruccions de dues vistes (veure apartat 2.7). Es va implementar *Bundle Adjusment* mitjançant la llibreria

Simple Sparse Bundle Adjusment. *Bundle Adjustment* va permetre visualitzar clarament que la reconstrucció resultant d'encadenar reconstruccions de dues vistes no és vàlida al crear núvols totalment desvinculats entre ells.

4.3.4 – Reconstrucció de més de dues càmeres correctament i *SIFT*.

El quart disseny modificava quasi completament tot el disseny de la part de reconstrucció, ja que el programa canviava totalment de filosofia. El nou programa implementa la solució vàlida comentada a l'apartat 2.7. A més a més, es va afegir la cerca de punts per *SIFT*.

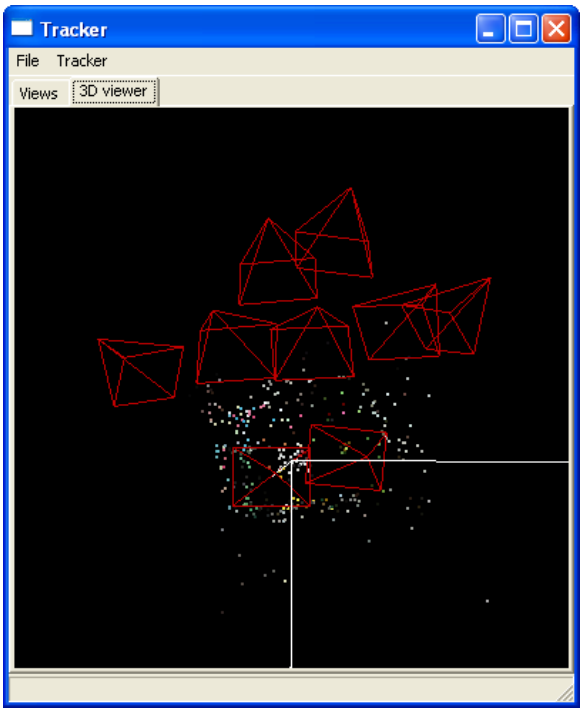


Fig. 22: Captura de la quarta versió del programa.

Part de la interfície també va canviar, ja que degut al nou mètode de reconstrucció (basat en la informació de múltiples imatges) ja no tenia sentit mostrar les rectes epipolars entre dues imatges. El programa es limitava a mostrar les vistes i la reconstrucció, tal i com es mostra a la figura 22

.4.4 – Diagrama de classes.

El diagrama de classes de l'últim disseny correspon a la figura 23. Descripció de les classes:

- *FeaturePoint*: s'encarrega d'emmagatzemar el punt 2D, la vista a que correspòn, el descriptor i l'índex del punt 3D corresponent.
- *FeatureDetector*: implementa els diferents algorismes de detecció de punts (*SIFT* i *SURF*).
- *View*: s'encarrega d'emmagatzemar tota la informació corresponent a la vista (imatge i càmera).
- *Camera*: emmagatzema tota la informació relativa a la càmera (paràmetres intrínsecs, rotació, translació...).
- *MotionStructure*: Consisteix en un conjunt de *View*, un *FeatureDetector* i un conjunt de *FeaturePoint*. Permet afegir imatges a l'estructura i reconstruir-les. Implementa tot el procés en les diferents funcions privades i genera un conjunt de punts 3D corresponent al núvol de punts.

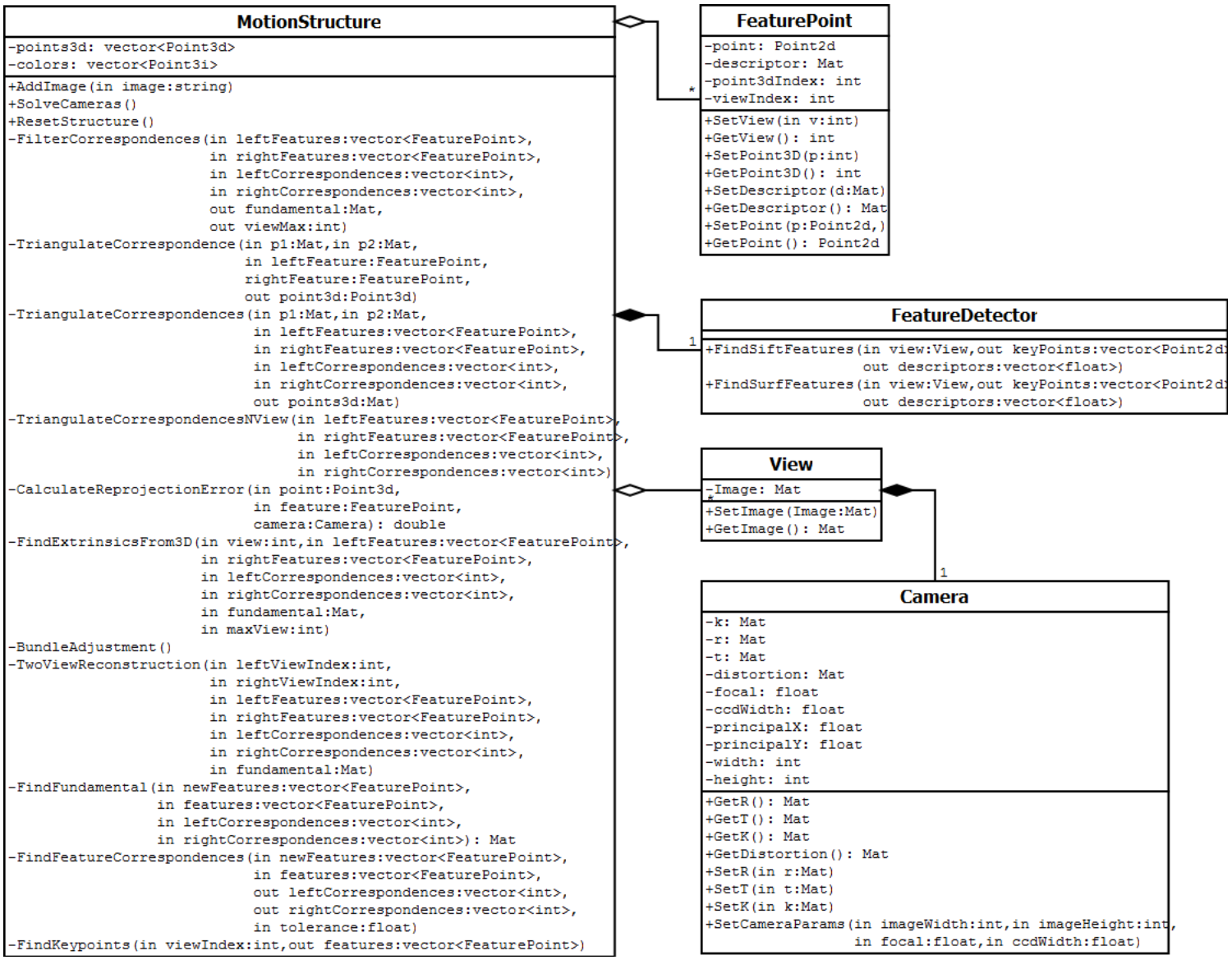


Fig. 23: Diagrama de classes de l'aplicació.

4.5 - Eines de desenvolupament.

El programa ha estat desenvolupat en *C++*. L'entorn de desenvolupament escollit ha estat *Eclipse CDT* i el compilador *gcc/g++*, ja que són eines de desenvolupament lliures i ja hi estava familiaritzat.

Llibreries utilitzades:

- OpenCV 2.0 [19]:

Llibreria lliure mantinguda per *Intel* que implementa multitud d'algorismes relacionats amb la visió per computador. El projecte fa ús del seu sistema de matrius, càlcul de la matriu Fonamental, *DLT*, detectors de punts *SURF*, arbres de cerca, *FAST*, *optical flow*...

El projecte va començar utilitzant la versió 1.2 i la versió 2.0 va ser alliberada durant el desenvolupament. Les millores de la versió 2.0 van ajudar molt al desenvolupament del projecte. Més tard va sortir també la versió 2.1, però algunes funcions que s'utilitzaven no eren del tot compatibles amb la versió 2.0 i es va optar per no actualitzar.

- WxWidgets 2.9 [18]: *Framework* multiplataforma usat per la creació de la interfície gràfica. S'ha optat per aquesta *framework* per facilitar una futura portabilitat a altres sistemes de forma més senzilla.

- OpenGL i GLUT [24]: Llibreria gràfica utilitzada per visualitzar dels resultats en 3D.

- Simple Sparse Bundle Adjusment [17]: Llibreria que implementa l'algorisme d'optimització *Bundle Adjusment* en C++ oferint molt bones prestacions.

- SIFT feature detector [16]: Llibreria que implementa l'algorisme *SIFT*.

Capítol 5. Resultats.

Els resultats que es presenten en aquest capítol són reconstruccions de col·leccions d'imatges de test d'altres projectes realitzats per especialistes en la matèria. Durant el desenvolupament del projecte s'han provat diferents seqüències de entre 2 i 40 imatges, produïnt reconstruccions de entre 2000 i 10000 punts.

Les reconstruccions que es mostren a continuació han estat obtingudes a partir de l'algorisme *SIFT*. L'algorisme *SURF* mostra bons resultats quan les imatges mostren poques diferències entre sí. En aquest cas tan *SIFT* com *SURF* mostren resultats pràcticament iguals. Per imatges on hi ha alts canvis d'escala i desplaçaments, *SIFT* supera amb escreix *SURF* permetent reconstruccions que no serien factibles mitjançant *SURF*. L'únic avantatge de *SURF* respecte *SIFT* és el temps de càlcul. *SURF* és força menys complex i més eficient reduint així el temps d'execució. Tot i així, avui en dia existeixen implementacions de *SIFT* que usen la *GPU* (Graphics Processing Unit) de la targeta gràfica

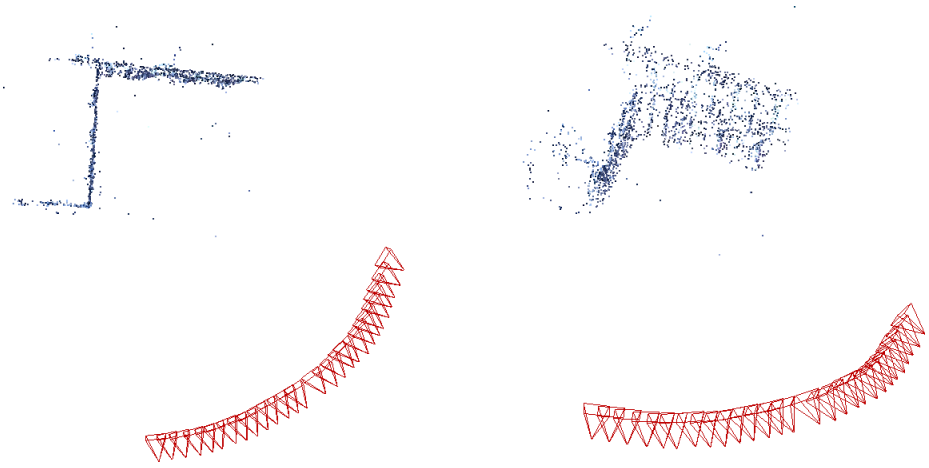
per tal d'accelerar l'execució de l'algorisme permetent així processament a temps real.

El principal problema que sorgeix al presentar resultats sobre aquest tema és trobar una forma de poder validar-los. Tan el núvol de punts com la posició de les càmeres és diferent per a cada programa, ja que les reconstruccions són ambigües en escala i les implementacions dels detectors de punts són diferents. Aquest fet descarta la possibilitat de comparar de forma rigorosa les reconstruccions obtingudes amb les d'altres programes. Una possible validació és comparar la reconstrucció obtinguda amb l'objecte original, però no sempre es disposa d'aquest objecte.

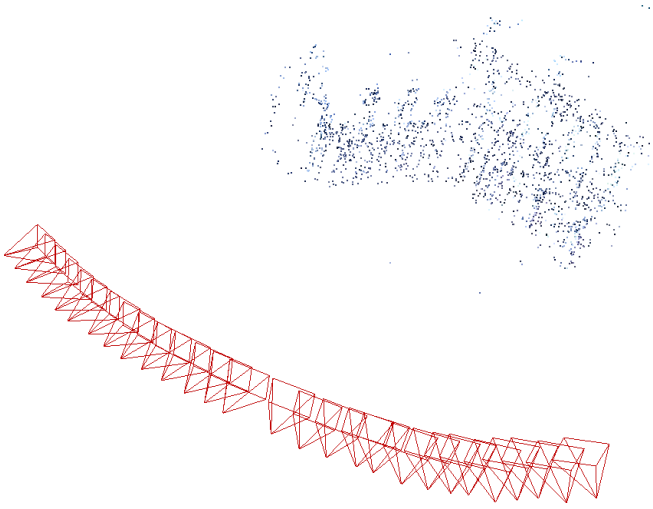
5.1 – Seqüència del castell i del dinosaure.

La figura 24 mostra la reconstrucció de 27 fotogrames d'un edifici. Les imatges han estat obtingudes a partir d'una càmera de vídeo. Al no disposar dels paràmetres intrínsecs de la càmera el resultat esperat era una reconstrucció projectiva. Com es pot observar a la figura 24 b), els angles entre les parets de l'edifici són coherents (90º), arribant a la conclusió que la reconstrucció és mètrica i no projectiva. L'autocalibració apareix degut a que pressuposem que els paràmetres intrínsecs de la càmera no varien. A mesura que s'afegeixen vistes a la reconstrucció l'algorisme *Bundle Adjustment* estima els paràmetres intrínsecs de la càmera fins a assolir la reconstrucció mètrica. Aquest mateix sistema permet la reconstrucció mètrica de la seqüència de la figura 25 on tampoc es coneixien els paràmetres intrínsecs.

a)

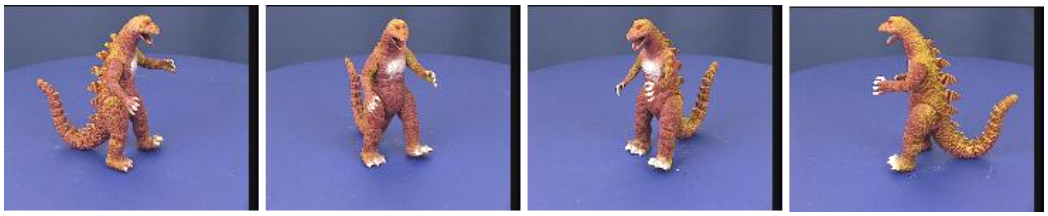


b) c)

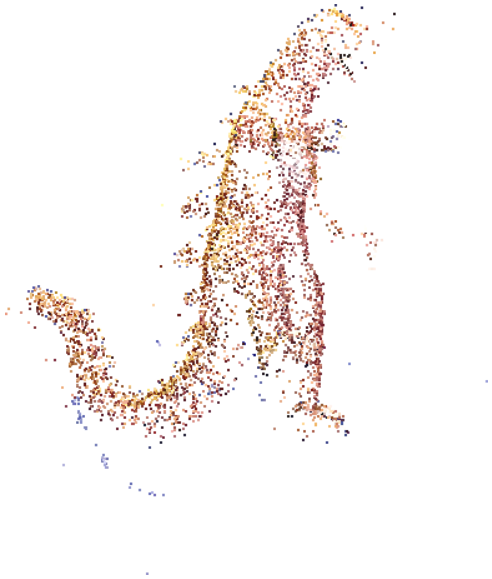


d)

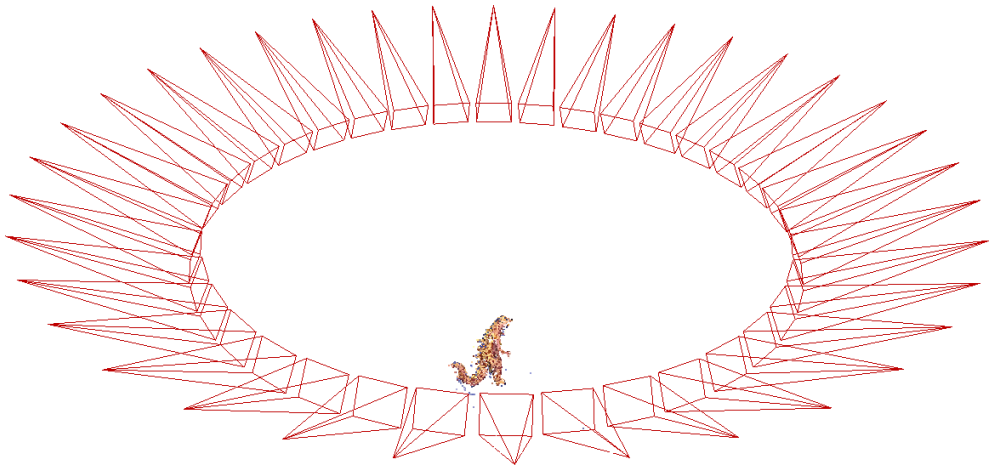
Fig. 24: a) mostra de les imatges d'entrada (proporcionades per [21]). b), c), d) Reconstrucció d'un castell vista des de diferents angles.



a)



b)



c)

Fig. 25: a) Mostra de les imatges d'entrada (proporcionades per [22]). b) Detall del núvol de punts obtingut. c) Detall de les 36 càmeres reconstruïdes.

5.2 – Seqüència de l'Hospital de Sant Pau.

La figura 26 mostra la reconstrucció de dues vistes de l'hospital de Sant Pau de Barcelona. A partir del núvol de punts, es va extreure una vista superior de la reconstrucció mitjançant una projecció ortogràfica dels punts. Aquesta vista es va sobreposar a una vista satèl·lit de l'edifici i es va adaptar manualment l'escala i la rotació de la vista dels punts per tal que encaixés amb la vista del satèl·lit de la millor manera possible. Com es pot observar a la figura 27, els punts reconstruïts s'adapten a l'estructura original mantenint les proporcions, angles i distàncies.

a)

b)

Fig. 26: a) Dues vistes de l'hospital de Sant Pau de Barcelona. b) Reconstrucció obtinguda.

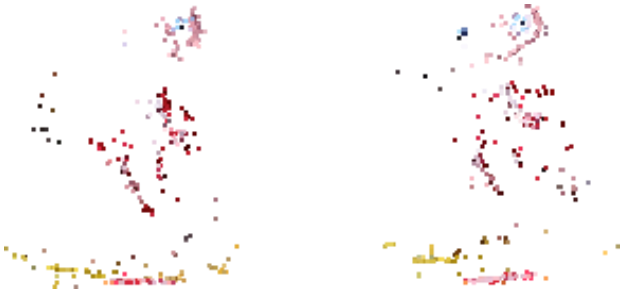
Fig. 27: Vista de satèl·lit de l'hospital de Sant Pau amb el núvol de punts de la reconstrucció sobreposat (punts verds).

5.3 – Validació dels paràmetres intrínsecs obtinguts.

Bundler [23] és el programa descrit a [15] que realitza reconstruccions a partir de col·leccions desordenades de fotografies. És el nucli de l'aplicació *Photosynth* [4]. *Bundler*, a diferència de nosaltres, no pressuposa que la càmera és la mateixa en totes les fotografies permetent la reconstrucció de multitud d'imatges allotjades a Internet. La figura 28 mostra una comparació entre el núvol de punts generat per *Bundler* i el nostre, confirmant que es difícil jutjar una reconstrucció a partir del núvol de punts. El núvol de punts ens ajuda, si més no, a veure si anem ben encaminats.

Tan el nostre programa com *Bundler* realitzen una estimació dels paràmetres intrínsecs de les càmeres. Com que nosaltres pressuposem que la càmera és constant, obtenim una sola distància focal. *Bundler*, al no fer aquesta suposició, calcula una distància focal per a cada càmera. Les distàncies focals que troba *Bundler* per la reconstrucció de la figura 28 són: 711.028, 685.664, 685.651, 687.016, 684.959 (5 imatges, distàncies en píxels). La distància focal obtinguda per nosaltres és 686.175. Com es pot observar, la distància focal obtinguda és molt similar a les distàncies focals que troba *Bundler*, podent afirmar així que la reconstrucció és bona.

a)



b) c)

Fig. 28: a) Mostra del conjunt d'imatges d'entrada. b) Reconstrucció obtinguda. c) Reconstrucció obtinguda mitjançant el programa *Bundler*.

Capítol 6. Conclusions i treball futur.

Aquest projecte presenta les fases inicials d'un sistema de conversió de seqüències d'imatges a 3D basat en la tècnica *structure from motion.* Ens hem centrat en resoldre el cas de la reconstrucció de càmeres calibrades per a col·leccions de fotografies. Hem començat per resoldre la reconstrucció de dues vistes i mitjançant *DLT* i *Bundle Adjustment* hem incorporat més càmeres a la reconstrucció de forma consistent.

Provant diferents algorismes detector-descriptor hem arribat a la conclusió que tan *SIFT* com *SURF* són adequats per trobar correspondències entre imatges, essent *SIFT* el més adequat al ser més tolerant a canvis d'escala, lluminositat i transformacions afins.

Hem tractat també la problemàtica de verificar els resultats, on hem aplicat validació per *ground truth* i validació dels paràmetres intrínsecs per comparació amb programes de l'estat de l'art.

El treball futur consisteix en els següents aspectes:

- Incorporar les millores proposades a [15] per tal de tenir unes reconstruccions més precises i robustes: no triangulació de punts a l'infinit, *Bundle Adjustment* local, incorporació de més d'una vista simultàniament...

- Realitzar la part de cerca de punts i correspondències a l'inici per tal d'obtenir un graf que expressi el veïnatge entre les fotografies i poder establir un odre automàtic d'incorporació d'imatges a la reconstrucció.

- Reconstrucció de seqüències de vídeo usant tensors trifocals i *tracking* 2D per *optical flow*.

- Implemetar mètodes d'autocalibració de seqüències de vídeo [20] per tal de poder reconstruir seqüències on els paràmetres intrínsecs no són constants.

- Rectificació d'imatges. La rectificació simplifica el procés d'obtenció de mapes de profunditat.

- Extracció de mapes de profunditat.

Glossari.

Structure from motion: Tècnica o procés de l'àmbit de la visió per computador que permet trobar l'estructura tridimensional d'una escena a partir del moviment de la càmera.

Geometria epipolar: Restriccions geomètriques entre dues càmeres que observen una mateixa escena.

Tracker 3D: Programa capaç de reconstruir el moviment d'una càmera a partir d'una seqüència de vídeo.

Mapa de profunditat: Imatge en escala de grisos on la intensitat de cada píxel es proporcional a la profunditat de la escena que representa.

Paràmetres intrínsecs: Paràmetres interns d'una càmera (distància focal, mida del sensor, coeficients de distorsió de la lent, punt principal...)

Paràmetres extrínsecs: Posició i orientació d'una càmera a l'espai.

Reconstrucció mètrica: Reconstrucció tridimensional que conserva les proporcions originals de l'escena.

Bundle adjusment: Algorisme d'optimització de reconstruccions que minimitza l'error de reprojecció refinant simultàniament els punts reconstruïts, els paràmetres extrínsecs i els paràmetres intrínsecs de les càmeres.

Direct Linear Transform (DLT): Algorisme que dedueix els paràmetres extrínsecs d'una càmera a partir de correspondències entre punts ja reconstruïts 3D i punts 2D de la imatge.

RANdom SAmple Consensus (RANSAC): Mètode iteratiu que estima paràmetres d'un model matemàtic a partir de mesures que contenen falsos positius.

Error de reprojecció: Diferència que hi ha entre un punt en una imatge i el seu punt 3D corresponent projectat.

SIFT: Algorisme de detecció i descripció de punts característics en una imatge que mostra molt bons resultats en canvis d'escala, lluminositat i transformacions afins.

SURF: Algorisme similar a SIFT però més eficient en quan a temps de còmput.

Bibliografia.

[1] R. Hartley and A. Ziserman, "Multiple View Geometry in Computer Vision," Second edition, Cambridge University Press, 2003.

[2] Pàgina principal de 2d3: http://www.2d3.com

[3] Pàgina principal de PFTrack: http://www.thepixelfarm.co.uk/

[4] Pàgina principal de Photosynth: http://photosynth.net/

[5] Pàgina principal d'ASIMO: http://world.honda.com/ASIMO/

[6] M.K. Chandraker, J. Lim and D.J. Kriegman. *Efficient Structure and Motion Using Lines.* ICCV 2009, Kyoto, Japan.

[7] J. Shi and C. Tomasi, "Good Features to Track'', Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94), pp. 593 - 600, 1994.

[8] David G. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, 60, 2 (2004).

[9] Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool, "SURF: Speeded Up Robust Features", Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346--359, 2008.

[10] Edward Rosten and Tom Drummond, "Fusing points and lines for high performance tracking", IEEE International Conference on Computer Vision (2005).

[11] Lucas B D and Kanade T, An iterative image registration technique with an application to stereo vision. Proceedings of Imaging understanding workshop, pp 121--130, 1981.

[12] Marius Muja and David G. Lowe, "Fast Approximate Nearest Neighbors with Automatic Algorithm Configuration", in International Conference on Computer Vision Theory and Applications (VISAPP'09), 2009.

[13] Informació sobre RANSAC: http://en.wikipedia.org/wiki/RANSAC

[14] Noah Snavely, Steven M. Seitz, Richard Szeliski. Photo Tourism: Exploring image collections in 3D. ACM Transactions on Graphics (Proceedings of SIGGRAPH 2006), 2006.

[15] Noah Snavely, Steven M. Seitz, Richard Szeliski. Modeling the World from Internet Photo Collections. International Journal of Computer Vision, 2007.

[16] Implementació de SIFT: http://web.engr.oregonstate.edu/~hess/

[17] Llibreria Simple Sparse Bundle Adjustment: http://www.cs.unc.edu/~cmzach/opensource.html

[18] Pàgina principal de WxWidgets: http://www.wxwidgets.org/

[19] Pàgina principal d'OpenCV: http://opencv.willowgarage.com/wiki/

[20] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, R. Koch, Visual modeling with a hand-held camera, International Journal of Computer Vision 59(3), 207-232, 2004.

[21] Pàgina principal de Marc Pollefeys: http://www.cs.unc.edu/~marc/

[22] Visual geometry group datasets: http://www.robots.ox.ac.uk/~vgg/data/data-mview.html

[23] Pàgina principal de Bundler: http://phototour.cs.washington.edu/bundler/

[24] Llibreria GLUT: http://www.opengl.org/resources/libraries/glut/

Firmat: Sergi Valls Company

Bellaterra, Setembre de 2010

Resum

Aquest projecte resol les fases inicials d'un altre projecte més gran que té com a objectiu la conversió automàtica de seqüències d'imatges a 3D. El projecte s'ha centrat en la reconstrucció calibrada de col·leccions d'imatges mitjançant la tècnica anomenada *structure from motion*. Aquesta tècnica forma part de l'àmbit de la visió per computador i s'utilitza per obtenir la posició i l'orientació de les diferents càmeres juntament amb una reconstrucció 3D de l'escena en forma de núvol de punts.

Resumen

Este proyecto resuelve las fases iniciales de otro proyecto mayor que tiene como objetivo la conversión automática de secuencias de imágenes a 3D. El proyecto se ha centrado en la reconstrucción calibrada de colecciones de imágenes mediante la técnica llamada *structure from motion*. Esta técnica forma parte del ámbito de la visión por computador y se utiliza para obtener la posición y la orientación de las distintas cámaras juntamente con una reconstrucción 3D de la escena en forma de nube de puntos.

Abstract

This project solves the previous stages of another bigger project which goal is the automatic conversion of image sequences into 3D. The project focuses on calibrated reconstruction of image collections using structure from motion computer vision technique. This technique allows us to recover the camera pose of all cameras together with a scene's sparse point cloud.